

Cited Reference

②

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-191872

(43)Date of publication of application : 28.07.1995

(51)Int.Cl.

G06F 11/28

G06F 9/46

G06F 11/32

(21)Application number : 06-308467

(71)Applicant : AT & T CORP

(22)Date of filing : 18.11.1994

(72)Inventor : FOWLER GLENN S
KORN DAVID G
KOUTSOFIOS ELEFTERIOS
NORTH STEPHEN C

(30)Priority

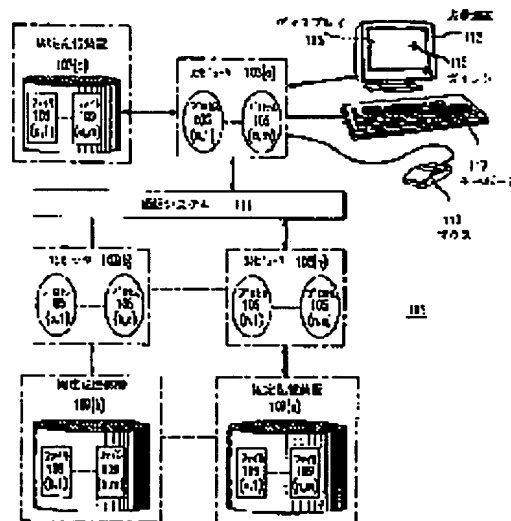
Priority number : 93 154771 Priority date : 19.11.1993 Priority country : US

(54) SYSTEM ANALYZING DEVICE

(57)Abstract:

PURPOSE: To provide a method for analyzing and controlling the operation of the system of a cooperative process.

CONSTITUTION: A system calling library used by this program executes system calling, and this is displaced with a dynamic link library which transmits a message indicating that the system calling is executed. The message is received by a display system 112. The display system 112 generates picture display indicating the present state of the system of a process in order to respond to this message. The picture display displays the system of the process like a tree. A node in the tree indicates the process in the system, and a resource such as a file used by this process. An edge in the tree indicates relation between the process and another process or the resource. A user can control which system calling generates the message, the speed of the display system required for responding to the message, and the execution of the process.



LEGAL STATUS

[Date of request for examination] 25.04.1997

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number] 2883826

[Date of registration] 05.02.1999

[Number of appeal against examiner's decision
of rejection]

[Date of requesting appeal against examiner's
decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平7-191872

(43)公開日 平成7年(1995)7月28日

(51)Int.Cl.⁸

G 0 6 F 11/28

9/46

11/32

識別記号

庁内整理番号

A 9290-5B

3 3 0 C 7629-5B

A 9290-5B

F I

技術表示箇所

審査請求 未請求 請求項の数27 F D (全 13 頁)

(21)出願番号 特願平6-308467

(22)出願日 平成6年(1994)11月18日

(31)優先権主張番号 1 5 4 7 7 1

(32)優先日 1993年11月19日

(33)優先権主張国 米国 (U S)

(71)出願人 390035493

エイ・ティ・アンド・ティ・コーポレーション

AT&T CORP.

アメリカ合衆国 10013-2412 ニューヨーク
ニューヨーク アヴェニュー オブ
ジ アメリカズ 32

(72)発明者 グレン ステファン ファウラー

アメリカ合衆国、07076 ニュージャージー、
スコッチ プレインズ、ライド プレイス 2322

(74)代理人 弁理士 三俣 弘文

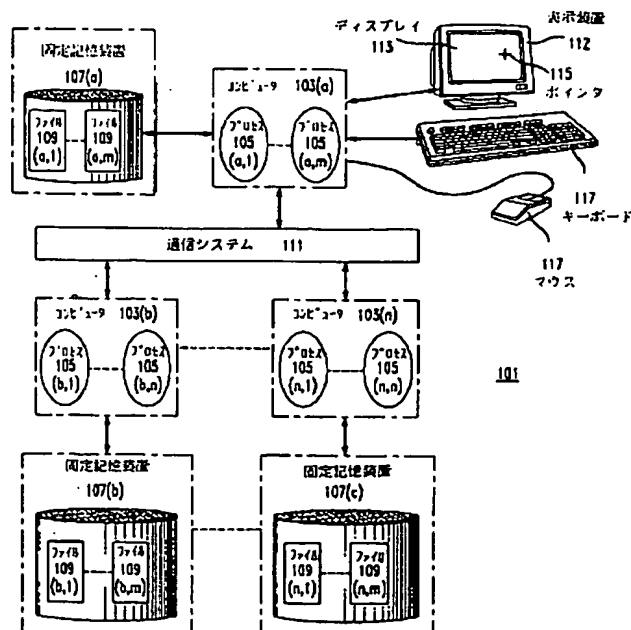
最終頁に続く

(54)【発明の名称】 システム解析装置

(57)【要約】

【目的】 共同プロセスのシステムの動作を解析・制御する方法を提供する。

【構成】 このプログラムにより使用されるシステム呼出のライブラリは、システム呼出を実行し、システム呼出が実行されたことを示すメッセージを送信する動的リンクライブラリにより置換される。メッセージは表示システムにより受信される。表示システムは、プロセスのシステムの現行状態を示す画像表示を生成することによりこのメッセージに応答する。画像表示はプロセスのシステムをツリー状に表示する。ツリー内のノードはシステム内のプロセスと、このプロセスにより使用されるファイルのような資源を示す。ツリー内のエッジはプロセスと他のプロセス又は資源との間の関係を示す。ユーザはどのシステム呼出がメッセージを生成するか制御でき、表示システムがメッセージに応答する速度を制御でき、プロセスの実行を制御できる。



【特許請求の範囲】

【請求項 1】 プロセスにより実行されるオペレーティング・システム呼出した結果としてのメッセージを、少なくとも一つのプロセスに送信させるモニタ手段と、プロセスのシステムの状態を示す画像表示を生成する手段と、からなることを特徴とするプロセスのシステム解析装置。

【請求項 2】 前記画像表示は、プロセスのシステムを有向非巡回グラフとして表示し、プロセス及びプロセスにより使用される資源はグラフ内のノードである請求項 1 の装置。

【請求項 3】 モニタ手段は、オペレーティング・システム呼出を行い、かつ、メッセージを送信するルーチンを更に含む請求項 1 の装置。

【請求項 4】 ルーチンは、プロセスにより実行されるプログラムと動的にリンクされる請求項 3 の装置。

【請求項 5】 モニタ手段は、ルーチンがメッセージをどのように処理するか決定するマスク手段を更に含み、オペレーティング・システム呼出を行うルーチンは、マスク手段により指示されるように、メッセージを更に処理する請求項 3 の装置。

【請求項 6】 マスク手段は、メッセージを送信すべきか否か示す第 1 のマスク手段を有する請求項 5 の装置。

【請求項 7】 マスク手段は、メッセージに関するフォームを示す第 2 のマスク手段を含む請求項 5 の装置。

【請求項 8】 モニタ手段は肯定応答メッセージを受信することができ、マスク手段は、送信されたメッセージを肯定応答すべきか否か示す第 3 のマスク手段を有し、オペレーティング・システム呼出を行うルーチンは、送信されたメッセージが肯定応答されるべき場合、送信されたメッセージは肯定応答されるべきであり、該肯定応答メッセージを待機中であることを示すことにより第 3 のマスク手段に応答する請求項 5 の装置。

【請求項 9】 各プロセスに対応するマスク手段が存在する請求項 5 の装置。

【請求項 10】 子プロセスはその親プロセスに対応するマスク手段のコピーを受信する請求項 9 の装置。

【請求項 11】 モニタ手段はマスクメッセージを受信することができ、かつ、送信されたメッセージをルーチンがどのように処理するか、マスクメッセージに応じて、マスク手段に決定させることによりマスクメッセージに応答することができる請求項 9 の装置。

【請求項 12】 メッセージに응答する手段がメッセージに응答する方法を制御する手段を更に含む請求項 1 の装置。

【請求項 13】 モニタ手段からメッセージを受信する手段を更に有し、メッセージに응答する手段はメッセージ受信手段からのメッセージを読み出し、メッセージ応答手段がメッセージに응答する方法を制御

する手段は、メッセージ応答手段がメッセージを読み出す方法を制御することにより制御を行う請求項 12 の装置。

【請求項 14】 装置のユーザに응答して、制御メッセージを送信する手段を更に有し、

モニタ手段は、制御メッセージにより必要とされるような挙動をプロセスにとらせることにより、制御メッセージに更に응答する請求項 1 の装置。

【請求項 15】 モニタ手段は、オペレーティング・システム呼出を行い、メッセージを送信するルーチンを更に含み、

ルーチンは実行を終了するために制御メッセージを待機している請求項 14 の装置。

【請求項 16】 モニタ手段は、送信されたメッセージを肯定応答すべきか否か示すマスク手段を有し、オペレーティング・システム呼出を行うルーチンは、送信されたメッセージが肯定応答すべきものである場合に、該送信メッセージは肯定応答すべきものであることを該メッセージ内に示すことによりマスク手段に응答し、

送信メッセージに응答する手段は、送信メッセージが肯定応答すべきものである場合に、その指示をユーザに供給することにより送信メッセージに응答し、

ユーザはこの指示に응答して制御メッセージを送信する請求項 15 の装置。

【請求項 17】 子プロセスを生成する場合又は資源にアクセスする場合、少なくとも一つのプロセスにメッセージを送信させる手段、及びこのメッセージに응答して、画像表示を発生させる手段からなり、

前記画像表示は有向非巡回グラフとしてプロセスのシステムを表示し、

プロセス及びプロセスにより使用される資源はグラフ内のノードであることを特徴とするプロセスのシステム解析装置。

【請求項 18】 グラフ内の第 1 のエッジは或る一つのプロセスと或る一つの資源との間の関係を示す請求項 2 又は 17 の装置。

【請求項 19】 グラフ内の第 2 のエッジは、最初の或る一つのプロセスと、この最初の或る一つのプロセスの子孫である次の或る一つのプロセスとの間の関係を示す請求項 2 又は 17 の装置。

【請求項 20】 システム内のプロセスは 1 個以上のプロセッサで実行し、所定の或る一つのプロセッサで実行するプロセス用のノードは所定のプロセッサを示すクラスタに属する請求項 2 又は 17 の装置。

【請求項 21】 ノードの形状は、ノードが或る一つのプロセス又は或る一つの資源を示すか否か指摘する請求項 2 又は 17 の装置。

【請求項 22】 資源はファイル資源と通信資源を含

み、

ファイル資源用のノードは第1の形状を有し、通信資源用のノードは第1の形状と異なる第2の形状を有する請求項21の装置。

【請求項23】 メッセージは一連のメッセージのうちの一つであり、

画像表示発生手段は、メッセージにより必要とされるように画像表示を変更する請求項2又は17の装置。

【請求項24】 コンピュータ内で実現され、かつ、第1のルーチンの第1のライブラリを使用するシステムの動作を解析するための解析装置であり、

第1のルーチンと同じ機能を果たし、更に、システムの変更を指示するメッセージを生成する第2のルーチンの第2のライブラリ、第1のライブラリ用の実行可能ノードを第2のライブラリ用の実行可能ノードと動的に置換する手段、第2のルーチンがメッセージをどのように生成するか指示する、第2のルーチンにアクセス可能なマスク手段、第2のルーチンは、該ルーチン内に指示されたようにメッセージを生成することによりマスク手段に20 応答する、メッセージに応答して、システムの状態を出力する手段、からなることを特徴とする解析装置。

【請求項25】 システムの状態はシステムの画像表示として出力される請求項24の解析装置。

【請求項26】 システムの画像表示は、有向非巡回グラフとしてシステムを表示し、システム内のエンティティは、グラフにノードとして現れ、エンティティ間の関係はグラフにエッジとして現れる請求項25の解析装置。

【請求項27】 解析装置のユーザが該装置用の制御メッセージを供給する手段を更に有し、

第2のライブラリはルーチンを受信する制御メッセージを含み、

第2のルーチンは制御メッセージに応答する請求項24の解析装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明はシステムの解析及び制御に関する。更に詳細には、本発明はコンピュータで実行するプロセスを構成するシステムの解析及び制御に関する。

【0002】

【従来の技術】 巨大なシステムを構築する際の重要な事項はデバッグである。すなわち、システムの実現においてエラーを検出し、解析し、そして訂正することである。システムがコンピュータで実行されるプログラムにより実現される場合、プログラムをデバッグするのに利用可能な様々なツールが存在する。ciaのようなディスカバリプログラムが存在する。

【0003】 このプログラムは、プログラマが、プログラムがどのように構成されているか理解するのを助け 50

る。このプログラムはデバッガと呼ばれる。このデバッガは、デバッグされているプログラムが実行されるときに起こる事象の内容をプログラマに見せることができる。最近のデバッガは、プログラマが、デバッグされているプログラムの実行を対話型で制御できるようになっている。

【0004】 このようなデバッガの一例は、フリーソフトウェア財団から入手できる、GDBデバッガである。プログラムの呼出履歴、分散メモリの並列プログラムにより発生されたイベント又はプログラムの並列実行のトレースなどのような情報を示すために、デバッガは更にグラフィック・インターフェースの使用を開始した。

【0005】 このようなグラフィック・インターフェースの一例は、IEEEコンピュータ、1993年6月発行、88-95頁に掲載された、アダム・ベグエリン(Adam Beguelin)らの、“複合環境におけるビジュアライゼーション及びデバッグ(Visualization and Debugging in a Heterogeneous Environment)”と題する論文に開示されている。

【0006】 ここで述べられているディスカバリプログラム及びデバッガはその本来の用途には完全に適合する。しかし、最近のシステムは一般的に、一連の共同サブルーチンとして実現されるのではなく、むしろ、一連の共同プロセスとして実現される。ここで、説明の便宜上、“プロセス”とは、ユーザ用のプログラムを実際に実行するコンピュータシステムにおける実体(エンティティ)として定義する。

【0007】 多くのシステムでは、共同プロセスは異なるコンピュータで実行する。システムが一連の共同プロセスとして実現される場合、システムのデバッグは、このプロセスにより実行されている個々のプログラムの理解とデバッグだけでなく、プロセスの共同動作の理解とデバッグも含む。このプロセスの共同動作の理解とデバッグは、ここに述べたプログラムディスカバリツール及びデバッガによっては行うことができない。

【0008】 現今のコンピュータシステムは、共同プロセスを構成するデバッグシステムのために貧弱な資源しか提供しない。例えば、UNIXオペレーティング・システム(UNIXはユニックス・システム・ラボラトリス社の登録商標である。)を使用するコンピュータシステムでは、オペレーティング・システムに対してプロセスにより為された呼出のリストを出力する“トレース”ユーティリティが存在する。

【0009】 また、どんなファイルが所定のプロセスをオープンするかユーザに伝える“オーファイル(ofil e)”ユーティリティ及びどんなプロセスが所定のファイルを使用しているか識別する“エフユーザ(fuser)”ユーティリティも存在する。これらの貧弱なデバッグツールの欠点は、単一のプロセッサで実行するプロセス

に関する情報しか提供しないことであり、従って、共同プロセスを異なるプロセッサで実行する理解及びデバッグシステムでは有用性が殆ど無い。

【0010】

【発明が解決しようとする課題】従って、本発明の目的は前記のようなシステムの精密な解析と制御を可能にする方法を提供することにより、共同プロセスを構成するデバッグシステムに伴う前記の問題点を解決することである。

【0011】

【課題を解決するための手段】本発明の方法は、ビジュアルプロセスマネージャで体现される。ビジュアルプロセスマネージャはモニタとグラフィカル・ディスプレイ・ジェネレータ（画像表示発生器）を有する。モニタは、システム内のプロセスが子プロセスを作成するか、又はファイルのような資源にアクセスするような操作に関するオペレーティング・システム呼出を実行する場合に、メッセージを送信する。

【0012】グラフィカル・ディスプレイ・ジェネレータはこのメッセージに回答して表示を生成する。この表示は、プロセスのシステムの状態及びメッセージにより必要とされるような変更を示す。従って、ユーザはプロセスのシステム内で起こっている事象を、この表示から決定できる。更に、ユーザは、この表示及びプロセスのシステムさえも制御するために、マウスのようなポインティング・デバイスを使用することもできる。

【0013】好ましい実施例では、この表示はノード及びエッジの有向非巡回グラフである。アクセスするプロセス及び資源はグラフ内のノードであり、ノードを接続するエッジはノード間の関係を示す。例えば、プロセスを示すノードはエッジによりその親用のノードに接続され、プロセスによりアクセスされるファイルに関するノードは別のエッジによりプロセス用のノードに接続される。

【0014】ビジュアルプロセスマネージャの重要な利点は、プロセスにより実行されるコードの変更が不要なことである。モニタは動的リンク・ライブラリにより実現される。動的リンク・ライブラリは、コンピュータシステムにより提供されたオペレーティング・システム呼出のライブラリを、システム呼出を行うだけでなく、メッセージも生成する新たなライブラリに置換する。

【0015】モニタ内のプロセス毎のマスクにより、ユーザは、どのシステム呼出がメッセージを生じるか、メッセージのフォームはどのようなものであるべきか、及びシステム呼出用のライブラリルーチンは継続前にユーザからの肯定応答を待つべきか否かを指示することができる。

【0016】好ましい実施例は、プロセスのシステムを解析するために本発明の方法を使用するが、本発明の方法は、一般的に適用可能であり、ルーチンのライブラリ

が、同じ機能を果たすが副作用としてメッセージも送信する動的リンク・ライブラリにより置換され得るような全ての状況において使用することができる。本発明の特別な利点は、システムを、システムのアプリケーションレベルコードの変更又はリコンパイルすること無しに解析できることである。

【0017】

【実施例】以下、図面を参照しながら本発明を具体的に説明する。

10 【0018】先ず、本発明を実現するシステムについて説明し、次いで、システムの好ましい実施例について説明し、最後に好ましい実施例の内容を具体的に説明する。

【0019】ビジュアルプロセスマネージャを有するシステム：図1

図1は常用の最新型分散コンピュータシステムの模式的構成図である。数台のプロセッサ103は通信システム111を介して相互に通信する。通信システム111は、メッセージをプロセッサ103間で送信できるものであればどのようなシステムであってもかまわない。各プロセッサの実行プログラムは多数のプロセス105である。

【0020】プロセス105はメッセージを、同一の又は他のプロセッサ103で実行する他のプロセスに送信できる。プロセス105は固定記憶装置107内のファイル109ともアクセスできる。ファイルは、プロセス105を実行するプロセッサ103に属する固定記憶装置107に属するか又は別のプロセッサ103に属する固定記憶装置107に属することができる。

30 【0021】ビジュアルプロセスマネージャのユーザは、表示装置112と、何れかのプロセッサ103（この場合は、プロセッサ103（a））に接続されたキーボード117及びマウス119のような入力装置を有する。キーボード117又はマウス119により、ユーザは表示装置112のディスプレイ上の表示に回答してプロセッサ103（a）に入力を送ることができる。更に詳細には、ユーザはマウス119を用いてディスプレイ113内のポインタ115を移動させることができ、また、マウス119のボタンを使用してコマンド入力を行うことができる。

【0022】ビジュアルプロセスマネージャのユーザビュー：図2、図3及び図5

ビジュアルプロセスマネージャのユーザは一般的に、図2に示されるようなディスプレイにより作業する。グラフ201は表示装置112上に現れる。大抵の場合、プロセッサ103はウインドウシステムを実行し、グラフ201はディスプレイ113内のウインドウ内に現れる。グラフ201は有向非巡回グラフである。

50 【0023】このグラフは4台のコンピュータで実行する共同プロセスのシステムを示す。各コンピュータは、

コンピュータの名称(例えば、キーウィ(kiwi))を有するクラスタ203で示される。各クラスタの内部是一群のプロセスボックス205である。各プロセスボックス205は、クラスタにより示されるコンピュータで実行するプロセス105を示す。プロセスボックス内の番号は、このプロセスに関するプロセス識別子(pid)である。

【0024】プロセス識別子は、プロセスを識別するためにコンピュータ203のオペレーティング・システムにより使用される番号である。文字列はプロセスにより現に実行されているプログラムの名称である。従って、pid23087を有するプロセス105はgccプログラムを実行する。?を有するプロセスボックス205のプロセス105はプロセス初期化プログラムを実行する。点線207は、プロセス105間の親子関係を示す。

【0025】親プロセスが他のプロセス105を生成するオペレーティング・システムコマンドを実行する場合、プロセス105は別のプロセス105の親である。従って、プロセス23086はプロセス23088の親である。

【0026】言うまでもなく、プロセス105はファイル109にアクセスできる。ファイルはファイル長円209としてグラフ201に現れる。各ファイル長円209は、ファイル長円209により示されるファイルの名称である文字列を有する。プロセス105がファイル109をオープンした場合、このプロセスのプロセスボックス205とこのファイルのファイル長円209を接続する実線のエッジ213が存在する。

【0027】プロセス105が読出用のファイルをオープンした場合、矢線の先頭はプロセスボックス205に存在する。プロセス105が書込用のファイルをオープンした場合、矢線の先頭はファイル長円209に存在する。読出用及び書込用の両方のファイルがオープンされた場合、矢線の先頭はエッジ213の両端に存在する。

【0028】プロセス105は、本発明が使用されるシステム内で、UNIXオペレーティング・システムにより提供されるパイプ及びソケットにより通信する。パイプは、同じコンピュータ103で実行するプロセス間の通信にのみ使用され、ソケットは、異なるコンピュータ103で実行するプロセス間の通信に使用される。ソケットの場合、通信は実際に通信システム111を介して伝わる。パイプ及びソケットは、パイプ環215及びソケット環211としてグラフ201に現れる。

【0029】ソケット環211は、ソケットが生成された時のソケットの受け手であったコンピュータの名称及び受け手コンピュータにおけるポートの番号を有する。ファイル長円209の場合と同じ方法で、エッジ213はパイプ環215及びソケット環211と一緒に使用される。すなわち、パイプ又はソケットが書込のためにオ

ープンされる場合、矢線の先頭は環211又は215に存在し、パイプ又はソケットが読出のためにオープンされる場合、矢線の先頭はプロセスボックス205に存在する。

【0030】好ましい実施例が実現されるUNIXオペレーティング・システムのバージョンでは、パイプは片方向であり、或るプロセスはパイプに書込み、他のプロセスはパイプから読出す。これに対して、ソケットは双方向である。

【0031】ビジュアルプロセスマネージャの動作：図3及び図5

好ましい実施例では、ビジュアルプロセスマネージャは2つの主要構成要素からなる。一つは、プロセスの実行に応じて、システム内のプロセスをモニタする構成要素であり、もう一つは、モニタ用構成要素からメッセージを受信し、このメッセージに応じてグラフ201を生成する構成要素である。

【0032】好ましい実施例における構成要素は、1993年に米国のシンシナティで開催されたUSENIX夏季会議講演集の279～290頁に掲載された、グレン・フォウラー(Glenn Fowler)らの“ユーザレベル複製ファイルシステム(A user-level replicated file system)”と題する論文に記載された、n次元ファイルシステム(nDFS)に基づくモニタと、プログラムdot(IEEEトランザクションズ・オン・ソフトウェア・エンジニアリング、Vol. 19, No. 3 (1993年3月)に掲載されたエムデン(Emden)らの“有向グラフの作図方法(A technique for drawing directed graphs)”と題する論文に説明されている)及びlefty(カナダ、アルバータ州のカルガリにおける91年度グラフィックス・インターフェース(Graphics Interface), 68～76頁(1991年)に掲載された、エレフシェリオス・コウトソフィオス(Eleftherios Koutsofios)らの“Lefty: テクニカルピクチャ用の2ビューエディタ(a two-view editor for technical pictures)”と題する論文中に開示されている)に基づくディスプレイジェネレータである。

【0033】好ましい実施例において、ビジュアルプロセスマネージャの動作を開始するために、ユーザはディスプレイジェネレータを実行する第1のプロセスと、モニタを実行する第2のプロセスをスタートさせる。各プロセスは表示装置112内にそれ自身のウィンドウを有する。第2のプロセスが実行されると、ユーザは、ログファイルを指定する、第2のプロセスのシェル内のセットアップコマンドを実行する。

【0034】モニタはこのログファイルにメッセージを送り、また、ディスプレイジェネレータはこのログファイルからその入力と、モニタにより監視されるべきオペレーティング・システムのシステム呼出を獲得する。好ましい実施例では、ユーザは、プロセス関連オペレーテ

ィング・システム呼出と、プロセス関連オペレーティング・システム呼出及びファイルアクセス関連オペレーティング・システム呼出、及び前記の二つと共に I/O システム呼出だけを指定できる。

【0035】好ましい実施例では、プロセス関連オペレーティング・システム呼出は UNIX フォーク (fork)、実行 (exec) 及びエグジット (exit) 呼出であり、ファイルアクセス関連呼出はオープン (open)、クローズ (close)、ダブ (dup) 及びパイプ (pipe) であり、I/O 呼出は読出 (read) と書込 (write) である。

【0036】前記のセットアップコマンドが実行されると、ディスプレイジェネレータ及びセットアップコマンドを実行するプロセスの現行状態により、モニタはログファイル読出へのメッセージ送信を開始し、そしてプロセスの子孫がグラフ 201 に表示される。従って、プロセスの所定のシステムをモニタするのに必要なことは、セットアップコマンドが実行されるプロセスの子としてシステムの第 1 のプロセスをスタートさせることだけである。

【0037】ディスプレイジェネレータはリアルタイムモードとシングルステップモードの 2 つのモードで動作する。リアルタイムモードでは、ディスプレイジェネレータは、モニタがメッセージをログファイル内に配置するのに応じて、ログファイル内のメッセージを読出す。シングルステップモードでは、モニタされるシステムは、モニタが終了するまで実行される。この場合、モニタからの全てのメッセージはログファイル内に配置される。

【0038】その後、ディスプレイジェネレータはログファイル内のメッセージを読出す。好ましい実施例では、モードは、ディスプレイジェネレータの実行を開始するコマンドのオプションにより指定される。いずれの場合も、グラフ 201 は、メッセージがログファイルから読出されるのに応じて、動的に更新される。

【0039】何れかのモードにおけるディスプレイジェネレータの動作は、ポインタ 115 がグラフ 201 のノード上に配置されず、マウスの右ボタンがクリックされた場合に、グラフ 201 に現われるグローバルメニューにより制御される。図 3 はリアルタイムモードに関するグローバルメニューを示す。メニューオプションは次の通りである。

【0040】●ドゥ・レイアウト (do layout): グラフ 201 のレイアウトをやり直す、

●リドロー (redraw): グラフをリフレッシュする、

●セーブ・グラフ (save graph): グラフをファイルにセーブする、

●ズームイン (zoom in): グラフを拡大する、

●ズームアウト (zoom out): グラフを縮小する、

●クリーンアップ (cleanup): デッドプロセスを示すノ

ードをグラフ 201 から削除する、

●ファインド・ノード (find node): ディスプレイをスクロールし、pid 又は名称により指定されるノードをグラフ 201 のディスプレイの中央に配置させる、

●スタート/ストップ (start/stop): アイテムが再び選択されるまで、グラフの更新を停止する、

●テキストビュー (textview): ディスプレイジェネレータにより現に解読されているプログラムを調べる、及び

●エグジット (exit): ディスプレイジェネレータの実行終了。

【0041】オプションはマウスを使用することにより選択される。すなわち、マウスでポインタ 115 をメニュー内の所望のエントリに移動させ、そしてマウスの左ボタンをクリックすることにより、オプションを選択できる。グラフ 201 が、グラフ 201 の表示されているウィンドウよりも大きい場合、ユーザはウィンドウ上のスクロールバーを使用し、グラフの非表示部分をウィンドウ内に持ってくることができる。

【0042】シングルステップモードのグローバルメニューは、スタート/ストップ及びクリーンアップが次の二つの選択肢により置換される点で、リアルタイムモードと異なる。

【0043】●プレイ・ノンストップ (play non-stop): ディスプレイジェネレータは停止することなくログファイル内のメッセージに応答する、

●プレイ・アンティル (play until): ユーザはシステム呼出名称又はメッセージ一連番号を指定する、ディスプレイジェネレータは指定されたシステム呼出名称又はメッセージ一連番号が到着するまでログファイル内のメッセージに応答する。

【0044】プレイ・ノンストップ又はプレイ・アンティルのどちらも選択されない場合、ユーザはメッセージがマウスにより読出される速度を制御する。ユーザがマウスの左ボタンをクリックすると、次のメッセージが読出され、グラフ 201 にそのメッセージに従って変更される。従って、シングルステップモードは、ユーザに、ユーザの関心のあるプロセスのシステムの動作部分だけを見せるオプションを与える。

【0045】グラフ 201 のノードに関するメニューにより、ユーザは、個々のノードに関する情報を発見することができ、また、プロセスボックス 205 の場合には、プロセスを操縦することができる。ノードメニューを得るために、ユーザはポインタ 115 を所望のノードに移動し、そしてマウスの右ボタンをクリックする。ノードメニューの内容はノードの種類及びディスプレイジェネレータがリアルタイムモード又はシングルステップモードで動作されているか否かに応じて変化する。

【0046】図 5 は、リアルタイムモードにおけるプロセスボックス 6678 に関するノードメニュー 501 を示す。可能なメニュー選択は次の通りである。

【0047】●プリント・エントリ(print entry)：ノードにより示される実体に関する情報を印刷する。プロセスノードでは、この情報はノード種類、プロセスが実行されるマシンの識別名(id)、マシンの名称、プロセスのpid、プロセスにより現に実行されているプロセスのパス名などである、

●プリント・アトリビュート(print attribute)：色、形状、フォント名及びフォントサイズなどのノードに関するディスプレイ属性を印刷する、

●キル(kill)：プロセスに対するUNIXオペレーティング・システムのキルコマンドを実行し、これを停止させる、プロセスはこのコマンドを無視することもできる、

●キル-9(kill-9)：プロセスを停止させるのに必要な形でキルを実行する、

●ランdbx(run dbx)：プロセスにより実行されているプログラム上でデバッグを実行させる。

【0048】シングルステップモードでは、プリント・エントリ(print entry)及びプリント・アトリビュート(print attr)の選択肢だけが利用できる。

【0049】ファイル長円209に関するメニューは、両方のモードにおいて、プリント・エントリ(print entry)及びプリント・アトリビュート(print attr)の選択肢だけを有する。プリント・エントリされる場合、ファイル名、デバイス名及びノード名が表示される。パイプ及びソケット環211に関するメニューはこれらの選択肢の他に、ショー・メッセージ(show message)選択肢も含む。

【0050】このショー・メッセージ(show message)選択肢は、モニタがI/Oシステム呼出を監視している時に、選択されたパイプ又はソケットを通過するメッセージを示す。パイプに関するプリント・エントリの場合、識別子が表示され、これにより、パイプを使用する各プロセスはそのパイプを知る。ソケットの場合、ソケットの名称、送信元マシン名、受信元マシン名及びポートが表示される。

【0051】実施例の具体的内容：図4、図6及び図7
図4は、ビジュアルプロセスマネージャの好ましい実施例の具体的構成を示す概要ブロック図である。ビジュアルプロセスマネージャ401は前記のようなモニタ418とディスプレイジェネレータ419の2つの主要構成要素を有する。モニタ418は、検査されているシステムに属するプロセス105の挙動を監視する。

【0052】ディスプレイジェネレータ419は、表示装置113上にディスプレイ201を生成し、キーボード117及びマウス119からの入力に応答する。構成要素間の通信はログファイル411を介して行われる。モニタ418は監視されているプロセス105から受信したメッセージをログファイル内に書き込み、ディスプレイジェネレータ419はログファイルからメッセージを

読出す。

【0053】モニタ418

モニタ418について更に詳細に説明する。モニタ418の構成要素は、検査されているシステムに属する各プロセス105用のメッセージ・ジェネレータ403と後置型サーバプロセス409からなる。プロセス105が、監視されているシステム呼出を実行する場合、メッセージ・ジェネレータ403はメッセージを生成する。

【0054】後置型サーバプロセス409はプロセス105からメッセージを受信し、ログファイル411にこのメッセージを書込む。プロセス105と後置型サーバ409との間の通信は、UNIXオペレーティング・システムにより提供されるソケットによる。

【0055】フォウラーの前掲書及び米国特許出願第08/0037号明細書に詳細に説明されているように、メッセージ・ジェネレータ403は、一連のライブラリルーチンに関する実行可能コードに各プロセス105を動的にリンクさせることにより実現される。監視されているプロセス105は、UNIXオペレーティング・システムにより提供される標準的なシステムに関するコードの代わりに、動的にリンクされたコードを実行する。

【0056】動的リンクはプロセスが初期化された時に起こる。実行可能コードはプロセス初期化時に動的リンクされるので、このプロセスにより実行されているアプリケーションコードを変更又は再リンクする必要が無い。動的リンクは、米国、カリフォルニア州のマウンテンビューに所在するサン・マイクロシステムズ(Sun Microsystems)社から1933に発行された“共用ライブラリ(Shared Libraries)”と題する文献中に詳細に説明されている。

【0057】所定のUNIXシステム呼出に関するライブラリルーチンにおけるコードはシステム呼出を呼出す。しかし、この他に、これは副作用も発生する。ビジュアルプロセスマネージャにおいて、この副作用はメッセージ406を包含する。図6はメッセージ406の詳細な内容を示す。このメッセージは次のような項目を含む。

【0058】●ソースpid(SPID)601：メッセージを生成したシステム呼出を行ったプロセス105のpid、

●ソースマシン(SMACH)603：プロセス105を実行するコンピュータの識別子、

●システム呼出名(Syscall name)605：プロセス105により行われたシステム呼出の名称、

●システム呼出引数(Syscall Args)607：プロセス105により行われたシステム呼出の引数、

●システム呼出結果(Syscall result)609：プロセス105により行われたシステム呼出の結果。

【0059】例えば、オープンシステム呼出に関するメ

ッセージ406では、引数607は、オープンすべきファイルのパス名及びファイルがオープンされる際のモードを含む。結果609は、新たにオープンされたファイルのデバイス番号及びノード番号を含む。

【0060】ビジュアルプロセスマネージャのユーザビュウの説明において指摘されるように、ユーザは、セットアップコマンドにおいて、どのタイプのシステム呼出を監視すべきか指定する。この情報は各プロセス105用のマスク405に記憶される。動的にリンクされたライブラリにおけるルーチンは、メッセージを送信する前にプロセス105用のマスク405をチェックし、マスク405が、メッセージ406をこの種のシステム呼出について送信すべきことを示す場合にのみ、メッセージ406を送信する。

【0061】好ましい実施例では、マスクはセットアップコマンドで設定され、システム呼出の種類について設定される。その後、マスクは監視されているプロセスの全てにより継承される。その他の実施例では、ユーザは、個々のプロセス及びシステム呼出についてマスクを設定するために、メニューを使用できる。好ましい実施例では、更に“簡潔(terse)”マスクを包含する。このマスクは、文字列をリターンするシステム呼出で指定できる。

【0062】簡潔マスクがシステム呼出について設定されない場合、システム呼出は文字列をメッセージ406にリターンする。簡潔マスクが設定される場合、システム呼出はポインタ115を文字列にリターンする。プロセスのマスクが設定されると、セットアップコマンドを介するか又は対話型で、プロセスの子孫により継承される。

【0063】ディスプレイジェネレータ419

ディスプレイジェネレータ419は、メッセージ解釈プロセス415とグラフ発生プロセス417の2つのプロセスとして実現される。好ましい実施例では、メッセージ解釈プロセス415は、コウトソフィオスらの前掲書に記載されたleftyプログラムを実行し、エムデンらの前掲書に記載されたdotプログラムを実行する。

【0064】メッセージ・インタープリタ415はログファイル411からメッセージ406を読み出す。リアルタイムモードでは、メッセージ・インタープリタ415は、後置型サーバ409がメッセージ406をログファイル411に書き込むのに応じて、メッセージ406を読み出す。シングルステップモードでは、メッセージ・インタープリタ415は、監視されているシステム内のプロセス105が全て終了した後だけ、ログファイル411からメッセージ406を読み出し始める。

【0065】各メッセージについて、メッセージ・インタープリタ415はメッセージ406からシステム呼出のタイプを決定し、グラフデータストラクチャ421(グラフ201を示す)で必要な変更をメッセージによ

り行う。メッセージ406がもはや存在しない場合、または最後のリドローの時から所定の数のメッセージ406が処理された場合、ディスプレイ201はリドローされる。グラフ201がリドローされる場合に起こる事象は、メッセージ406により必要とされる変更の種類により左右される。

【0066】変更がグラフ201の新たなレイアウトを必要としない場合、メッセージ・インタープリタ415はグラフデータストラクチャ421から情報を単に獲得し、この情報をウインドウシステム427に供給する。ウインドウシステム427はこの情報を使用し、表示装置112にグラフ201を実際に表示する。変更が新たなレイアウトを必要とする場合、メッセージ・インタープリタ415は新たなレイアウトに必要な情報をグラフデータストラクチャ421から獲得し、この情報を論理グラフ記述423として、グラフジェネレータ417に供給する。

【0067】グラフジェネレータ417は、論理グラフ記述に基づき物理グラフ記述425をメッセージ・インタープリタ415にリターンする。その後、メッセージ・インタープリタ415は、新たな物理グラフ記述をグラフデータストラクチャ421に組み込み、更新されたグラフデータストラクチャ421からの情報を前記のようにウインドウシステム427に供給する。

【0068】図7は、グラフデータストラクチャ421の詳細な構成図である。グラフデータストラクチャ421はノードアレイ701とエッジアレイ713の2つの主要構成要素を有する。ノードアレイ701はグラフ201内の各ノードに関するノードエントリ703を包含する。エッジアレイ713はグラフ201内の各エッジに関するエッジエントリ715を包含する。各ノードエントリ703は、下記の4種類の情報を包含する。

【0069】●論理ノード情報、メッセージ・インタープリタ415はこの情報を論理グラフデータ423としてグラフジェネレータ417に供給する、

●ノードディスプレイ情報707、グラフジェネレータ417はこの情報を、論理グラフデータ423に応答して物理グラフデータ425としてメッセージ・インタープリタ415に供給する、

●ビジュアルプロセスマネージャ情報709、及び

●エッジポインタリスト711。

【0070】ビジュアルプロセスマネージャ情報709は、ディスプレイ201内のノード、及びノードエントリ703により示されるノードに付随するメニューに現れる情報である。エッジポインタリスト711は、ノードエントリ703により示されるノードで開始又は停止するエッジに関する各エッジエントリ715に対するポインタを包含する。

【0071】エッジエントリ715は各エッジに関する同じ種類の情報を包含する。論理エッジ情報717は、

メッセージ・インタープリタ 415 がグラフジェネレータ 417 に供給したエッジに関する情報を包含する。エッジディスプレイ情報 719 は、グラフジェネレータ 417 によりメッセージ・インタープリタ 415 にリターンされる情報を包含する。VPM 情報 721 は、エッジに付随するメニューに現れる情報である。

【0072】ポインタ 723 から (from ptr 723) は、エッジの送り手であるノードアレイ 701 内のノードエントリ 703 に対するポインタである。ポインタ 725 へ (toptr 725) は、エッジの受け手であるノードアレイ 701 内のノードエントリに対するポインタである。

【0073】好ましい実施例で実現されるようなモニタ 418 の限界は、メッセージ・インタープリタ 415 が、プロセス 105 が停止したか否かメッセージ 406 から常に予測できるとは限らないことである。好ましい実施例では、この問題は、メッセージ・インタープリタ 415 に、所定のプロセス 105 から最後のメッセージ以来の時間を追跡させることにより処理される。

【0074】或る一定の制限時間を越えた場合、メッセージ・インタープリタ 415 は、所定のプロセスが実行されるコンピュータについて pid サーバ 429 に照会し、プロセス 105 が依然として活動中であるか否か決定する。プロセス 105 が活動中でない場合、メッセージ・インタープリタ 415 は、プロセス 105 に関する情報をグラフデータストラクチャ 421 から削除する。そして、グラフ 201 の次のリドロウの時に、プロセス 105 に関するノード及びプロセスにより単独でアクセスされるファイル又はパイプに関するノードが消失する。

【0075】更に、メッセージ・インタープリタ 415 は、ウインドウシステム 427 により供給されるような、キーボード 117 及びマウス 119 からの入力を処理する。例えば、ユーザがマウスの右ボタンをクリックし、グローバルメニュー又はノードメニューを要求すると、メッセージ・インタープリタ 415 は、要求されているメニューのタイプを決定し、このメニューに関する情報を、ウインドウを表示するウインドウシステム 427 に供給する。

【0076】同様に、ユーザが、情報の表示を必要とするメニューエントリを選択する場合、ウインドウシステム 427 は、この選択が行われたことを示し、メッセージ・インタープリタ 415 は、グラフデータストラクチャ 421 からこの情報を取得し、この情報を表示するためにウインドウシステム 427 に供給する。ユーザが特定のプロセスに関するノードメニューについてキルエントリを選択する場合、メッセージ・インタープリタ 415 は、プロセス 105 を削除するシステムコマンドを実行する。

【0077】ビジュアルプロセスマネージャ 401 の対話型バージョン

好ましい実施例では、端末におけるユーザはログファイル 411 内のメッセージがメッセージ・インタープリタ 415 により読出される速度を決定することだけができる。ユーザはビジュアルプロセスマネージャ 401 によりシステム内のプロセスの実行を制御できない。

【0078】別の実施例では、ユーザはプロセスのシステムの実行を実際に制御するために、ビジュアルプロセスマネージャ 401 を使用できる。この実施例では、ログファイル 411 はソケット 420 により置換される。ソケット 420 は後置型サーバ 409 をメッセージ・インタープリタ 415 へ直接接続する。更に、双方向性ソケット 416 は、監視されているプロセス 105 用のメッセージ・ジェネレータ 403 と後置型サーバ 409 とを接続する。

【0079】メッセージ 406、マスク 405 及び動的リンクライブラリ用のコードでも変更が必要である。メッセージ 406 は追加フィールドの ACK (肯定応答) フィールド 611 を帯びる。この ACK フィールド 611 は、メッセージ・インタープリタ 415 が後置型サーバ 409 から受信されたメッセージを肯定応答しなければならないか否か示す。マスク 405 は肯定応答マスクを包含する。肯定応答マスクは、システム毎の呼出ベースで、どのシステム呼出が肯定応答を必要とするか示す。

【0080】最後に、動的リンクライブラリルーチンは、メッセージの送信前に肯定応答マスクを検査するコードを含むように変更される。マスクが、システム呼出が肯定応答を必要とすることを示す場合、システム呼出を置換するライブラリルーチンはメッセージ内の肯定応答フィールド 611 をセットし、そして、後置型サーバ 409 からの肯定応答メッセージを待機する。メッセージが到着した後にだけ、ライブラリルーチンはその実行を完了する。

【0081】対話型モードの動作について説明する。プロセス 105 を対話型モードにするために、ユーザは前記のようなマウスを使用しプロセスのメニューを取得する。メニュー 501 は対話型モードを指定する追加エントリを有する。ユーザがこのエントリを選択すると、別のメニューが現れる。この別のメニューにより、ユーザは、どのシステム呼出をこのプロセスについて肯定応答すべきか指定できる。メッセージ・インタープリタ 415 はメニューから入力を獲得し、どのシステム呼出は現に肯定応答されているか示すノードのためのフィールドを設定する。

【0082】メッセージ・インタープリタ 415 はまた、ソケット 420 を介してメッセージを後置型サーバ 409 に送信する。このメッセージはシステム呼出が肯定応答されるべきプロセス 105 及び肯定応答されるべきシステム呼出を指定する。その後、後置型サーバ 409 はソケット 416 を介して、メッセージを指定プロセ

ス 105 に送信し、メッセージ・ジェネレータ 403 内のコードは、メッセージにより必要とされるような肯定応答マスクを設定することによりコードに応答する。

【0083】肯定応答マスクが設定された時点から、システム 401 は次のように動作する。プロセス 105 がシステム呼出を実行すると、このシステム呼出用のライブラリルーチンはシステム呼出用の肯定応答マスク 405 をチェックする。マスクが設定されていれば、メッセージ 406 はその肯定応答フィールド 611 設定を有し、ライブラリルーチンはシステム呼出の実行及びメッ

10 セー 406 の送信後、休止する。
【0084】メッセージがソケット 416、後置型サーバ 409 及び及びメッセージ・インタープリタ 415 内のソケット 420 から到着すると、メッセージ・インタープリタ 415 は、メッセージ 406 により必要とされるようなグラフ 201 の変更及びディスプレイ内のプロセス用のプロセスボックス 205 の外観の変更により、設定肯定応答フィールド 611 に応答し、プロセスボックス 205 により示されるプロセス 105 は肯定応答を待機中であることを示す。

【0085】ユーザはポインタ 115 をプロセスボックス 205 に移動させ、マウスの左ボタンをクリックすることにより、肯定応答を与える。ポインタ 115 の位置及びマウス左ボタンのクリックにตอบสนองして、メッセージ・インタープリタ 415 はソケット 420、後置型サーバ 409 及びソケット 416 を介して肯定応答メッセージをこのプロセス用のメッセージ・ジェネレータ 403 に送信する。このメッセージにตอบสนองして、ライブラリルーチンはその実行を終了する。これが行われている最中に、メッセージ・インタープリタ 415 は更にグラフ 201 の外観を変更し、プロセスはもはや肯定応答を待機していないことを示す。

【0086】前記の説明から明らかなように、システム 401 のユーザは、プロセス 401 の実行を任意の所望の粒度にまで制御することができる。その他の実施例も、肯定応答マスクのグローバル設定を可能にするグローバルメニュー 301 内にエントリを有することができる。

【0087】結論

以上、本発明のビジュアルプロセスマネージャの実現に関する好ましい実施例について説明した。しかし、前記のように、ビジュアルプロセスマネージャの実現に使用される技法は、プロセスのシステム解析に限定される手段ではなく、システムにより使用されるライブラリが別のライブラリにより動的に置換される全てのシステムで使用することもできる。

【0088】当業者には明らかなように、前記のビジュアルプロセスマネージャを実現する方法は他にも多数存在する。例えば、前記に説明した動的リンク技法及びグラフ作図技法は本発明の実現に特に有用であるが、モニ

タ 418 内のライブラリルーチンは、ディスプレイジェネレータ 419 を実現するのに使用できる lefty 及び dot プログラム以外のプロセス及びグラフ作図プログラムにより実行されるコードに静的にリンクさせることもできる。

【0089】監視すべきオペレーティング・システム呼出の選択は、オペレーティング・システム毎に変化し、また、ビジュアルプロセスマネージャの企図される用途に応じても変化する。同様に、利用可能なマスキングの種類は実現される対象毎に異なる。

【0090】同一の系統では、好ましい実施例におけるグラフ 201 で使用される形状、ラベル及び色彩が特に有用であるが、他の実施例では、他の形状及び他の色彩も使用でき、更に、どのような情報をノードに直接示すか、及び何がメニューによりアクセス可能かなどに関して異なる決定を行うこともできる。また、他の実施例では、ユーザに、ディスプレイ 201 の制御又はプロセスのシステムの操作の制御を多少は提供することもでき、システムの実行を開始させるために異なる方法を使用することもできる。

【0091】

【発明の効果】以上説明したように、本発明の方法によれば、コンピュータで実行するプロセスを構成するシステムの精密な解析と制御が可能になる。また、共同プロセスを構成するデバッキングシステムにおいて資源を有効に活用することができる。また、本発明の方法はシステム解析の他に、ルーチンのライブラリが、同じ機能を果たすが副作用としてメッセージも送信する動的リンクライブラリにより置換され得るような全ての状況において使用することができる。本発明の方法の特別な効果は、システムのアプリケーションレベルコードの変更又はリコンパイルすること無しに、システムを解析することができることである。

【図面の簡単な説明】

【図 1】本発明を実現することができる分散型コンピュータシステムの一例の模式的構成図である。

【図 2】本発明により生成された表示の一例の模式図である。

【図 3】本発明により生成された表示の別の例の模式図である。

【図 4】ビジュアルプロセスマネージャの好ましい実施例の構成を示す概要ブロック図である。

【図 5】本発明により生成された表示の更に別の例の模式図である。

【図 6】本発明で使用するメッセージのブロック図である。

【図 7】本発明で使用するグラフデータストラクチャのブロック図である。

【符号の説明】

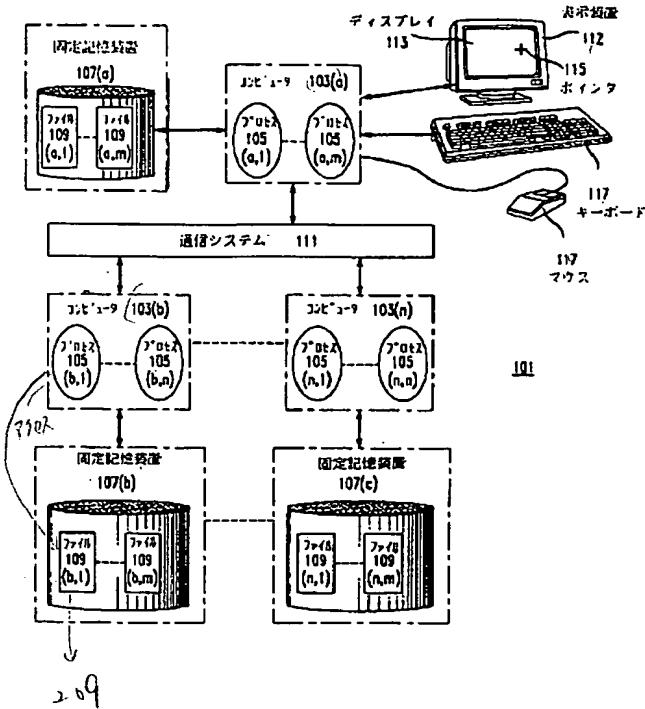
103 プロセッサ

105 プロセス
107 固定記憶装置
109 ファイル
111 通信システム
112 表示装置
113 ディスプレイ
115 ポインタ
117 キーボード
119 マウス

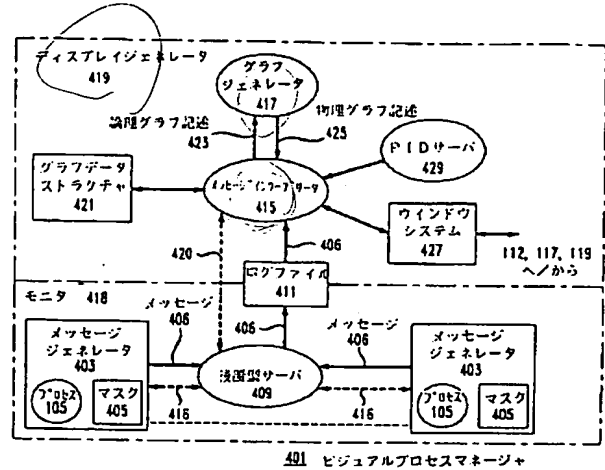
* 113 ディスプレイ
115 ポインタ
117 キーボード
119 マウス

*

【図 1】



【図 4】

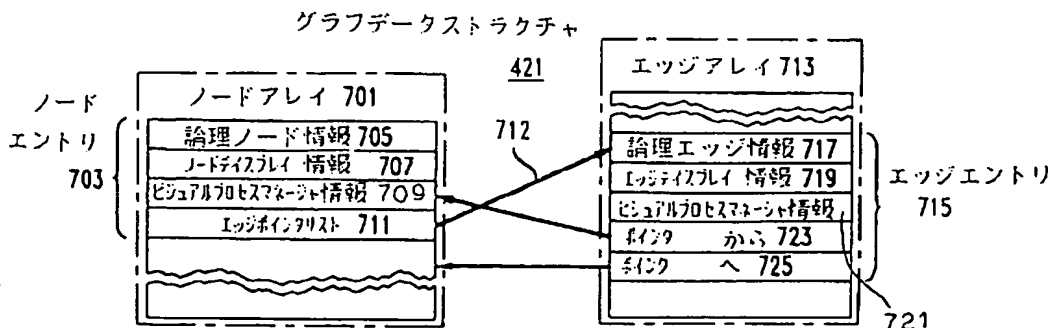


【図 6】

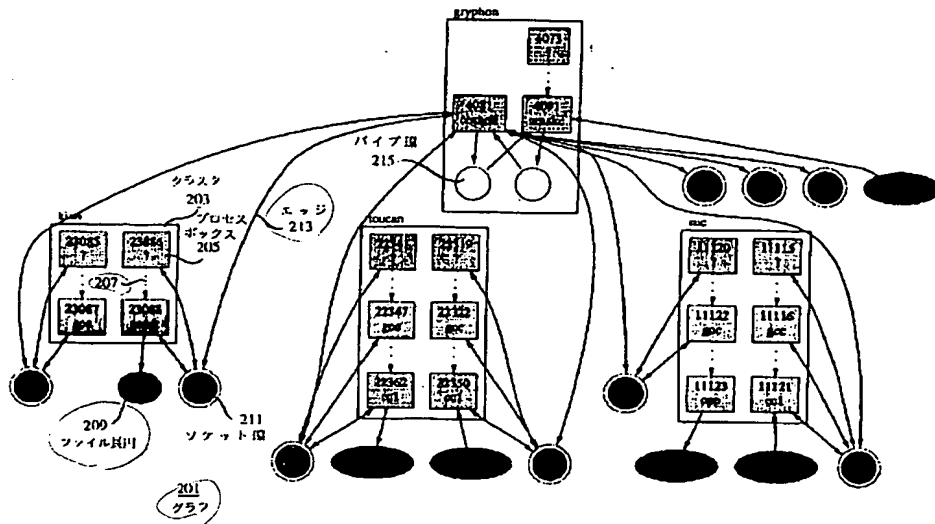
ソースid	ソースマシン	システム呼称	システム呼称	システム呼称	システム呼称
601	603	605	607	609	611

406

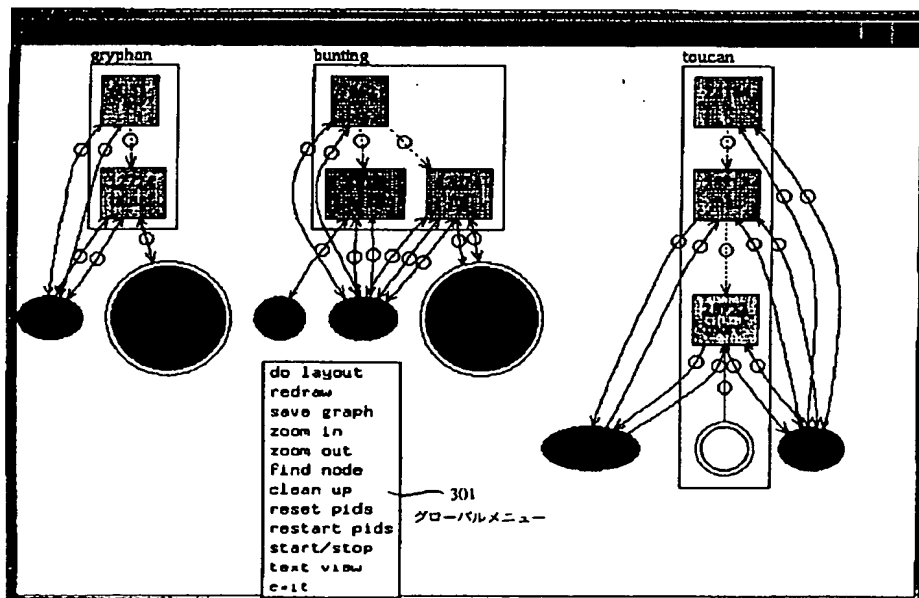
【図 7】



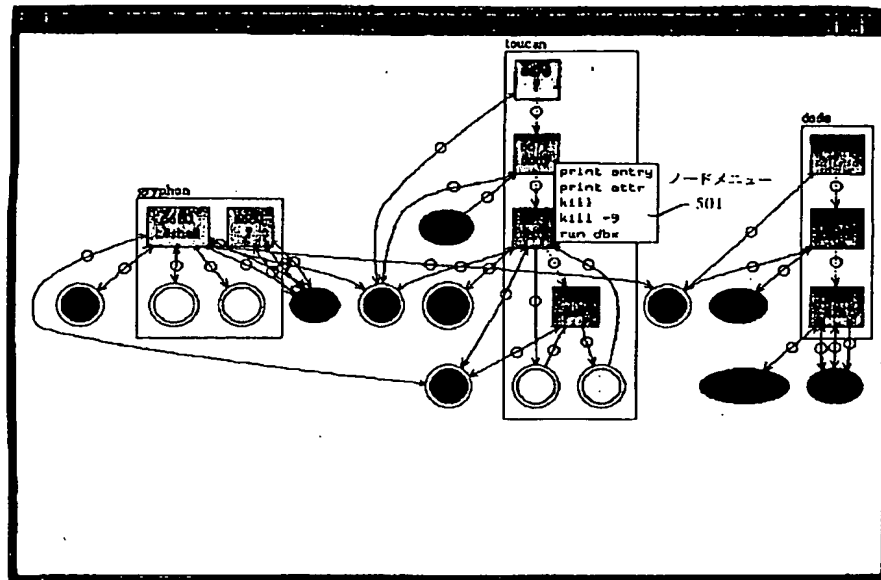
【図2】



【図3】



【図5】



フロントページの続き

(72)発明者 デビッド ジェラード コーン
 アメリカ合衆国、10003 ニューヨーク、
 ニューヨーク、エー 107、マーサー ス
 トリート 303

(72)発明者 エレフテリオス コウツオフィオス
 アメリカ合衆国、07928 ニュージャージ
 ー、チャタム、アプト. イー4、リバー
 ロード 420
 (72)発明者 ステファン シー. ノース
 アメリカ合衆国、07830 ニュージャージ
 ー、カリフォン、フィルハウアー アベニ
 ュー 127